## Software Review Report

### Quality Assurance Development and Methodology

Throughout the development process this software has maintained quality through many different processes and approaches, from multiple team members. During Sprint One, the team used the CITE Managed Services Quality Assurance standards as a starting point for good quality assurance practice. This was summarised in the ‘Analysis Report’; building on the previously created prototype and employing these quality assurance practices as standard across each Sprint of the project development. In addition to this, testing was carried out comprehensively on all aspects and features of the application as documented in the ‘Test Plan’ and subsequent sprint testing documentation, test tables and screen captures, ensuring good UX, UI and backend functionality. This testing was also implemented with the new features required for each Sprint such as; adaptive/responsive, multi-platform, performance, user groups and administrator access, optimisation, the rating system and google charts display.

During Sprint Two, quality assurance was ensured by the team conducting a software review plan, which detailed the measures undertaken to provide excellent quality assurance including code review, team demonstrations, code testing, client feedback, and product demonstration to the client over the three sprint cycles. Sprint three further confirmed high QA by providing an optimisation report detailing the improvement of the product using code review and optimisation tools. In addition, the PHP code sniffer tool has been used to validate all php scripts to the PEAR standard. By providing all these measures with the final product in addition to the supporting documentation, quality assurance is shown to have been maintained throughout the delivery of the movie database application.

### Testing Methodology

Each testing process uses the same blueprint to carry out testing. This process is detailed in the ‘Test Plan’ and can be summarised as follows. Testing is carried out on two levels;

* to comply with the CITE standards and to also ensure the clients requirements and requests are met with satisfaction
* testing that the application functions as required and as intended, securely and without errors

These two levels are met by carrying out regular verification on both frontend and backend code, the database, and on user experience using use cases. When bugs are detected they will be classified in order of their impact on the applications function. Bugs which negatively impact the user experience or execution of code function will be prioritised.

This blueprint is followed by the tester creating a test document consisting of a test table which carries out an example of every possible user input, action or function. This ensures complete test coverage and each use case is to be documented with screen capture evidence. Testing is considered complete when it has been carried out on each function and non-functional requirement, any identified issues (by either team member or client) have been addressed and resolved, and testing documentation has been completed in full.

### Future Support

Due to the highly documented nature of the delivery of this project, future development would be very straightforward. Any user, or any person with expertise could read through the provided documentation and development plans to ascertain an understanding of the nature of how things work. Each script, webpage, and CSS files have comment blocks which explain the functions carried out in each. The final handover project folder also includes a document called ‘RADprojectFileMap’ which describes how each page of the application runs in terms of all the different components i.e. CSS, scripting, PHP etc.

Furthermore, the source control for this project could be altered to allow access, or a version could be supplied to the client for support and backed up in case of data loss/corruption. This allows for future modifications to regress to previous sprints if there was an unforeseen change in the future that required this, or a bug that cannot be fixed.

### User Requirements Mapping

The following diagram shows how different types of users would interact with the application. Each user should be able to perform intended functions with relative ease and get the intended results, with an interface that is easy to navigate, understand and even intuitive of their needs. The requirements for staff and administrators are different; as these users provide support and facilitate the backend operations for the end users to find results by maintaining an up-to-date database, security, and subscription services. The final product delivered by the team satisfies all of these requirements while also being highly accessible, easy to understand and relevant.

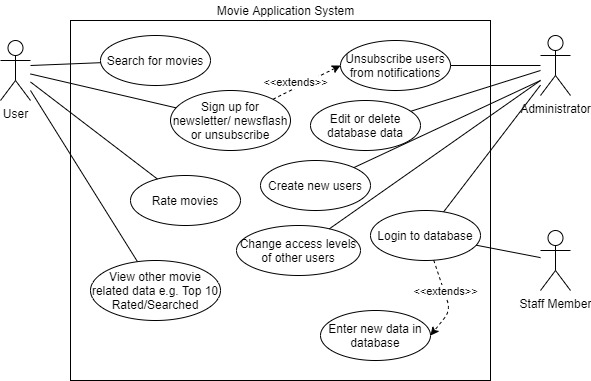


Figure : A Use Case diagram represents how different users interact with the Movie database System.